DSP505: Programming Lab for Data Science and Artificial Intelligence

TPL616: Advanced Programming for DSAI

Lab-1: Practice Problems

Due Date: 9-Aug-2025

Question-1:

Given two Python lists, iterate over both lists simultaneously such that list1 should display items in original order and list2 in reverse order. Once the smaller list exhausts, the other list elements should be printed alone.

Input:
```
list1 = [10, 20, 30, 40]
list2 = [1, 10, 100, 200, 300, 400]
```

Output
```
10 400
20 300
30 200
40 100
10
1
```

Question-2:

Given a 2D board and a word, find if the word exists in the grid. The word can be present horizontally or vertically.

Example:

Input:
```
2D Board:
[
  ['A','B','C','E'],
  ['S','F','C','S'],
  ['A','D','E','E']
]
Word: FD
```

Output: True

Question-3:

You are given an n x n 2D matrix representing an image. Rotate the image by 90 degrees (clockwise) in-place.

Example:

Input:

```
matrix = [
  [1,2,3],
  [4,5,6],
  [7,8,9]
]
```

Output:

```
[
 [7,4,1],
 [8,5,2],
 [9,6,3]]
```

Question-4:

Write a function to determine if a 9x9 Sudoku board is valid. Only the filled cells need to be validated according to Sudoku rules.

Input:

```
 [
  ["5","3",".",".","7",".",".",".","."],
  ["6",".",".","1","9","5",".",".","."],
  [".","9","8",".",".",".",".","6","."],
  ["8",".",".",".","6",".",".",".","3"],
  ["4",".",".","8",".","3",".",".","1"],
  ["7",".",".",".","2",".",".",".","6"],
  [".","6",".",".",".",".","2","8","."],
  [".",".",".","4","1","9",".",".","5"],
  [".",".",".",".","8",".",".","7","9"]
]
```

Output: True

DSP505: Programming Lab for Data Science and Artificial Intelligence

TPL616: Advanced Programming for DSAI

Lab-2: Practice Problems (OOP)

Due Date: 30-Aug-2025

Task-0:

Execute the example codes discussed in the class to make yourself familiar with the classes and objects.

Task-1:

Assume that you are hired as a software developer by an e-commerce company (like Amazon/Flipkart) to design the backend shopping cart system. Customers will browse products, add them to their shopping cart, and place orders. You need to implement the core classes and logic using Object-Oriented Programming principles in Python.

1. Product Class

Attributes:

- - product_id, name, price, stock (number of items available)

Methods:

- - update_stock(quantity) : decreases/increases stock
- - is_available(quantity): checks if requested quantity is in stock

2. CartItem Class

Represents a product inside the cart.

Attributes:

- - product (Product object), quantity (number of items added to cart)

Methods:

- - get_total_price() → returns price * quantity

3. ShoppingCart Class

Attributes:

- - cart_items (list/dict of CartItem objects)

Methods:

- - add_product(product, quantity)
- - remove_product(product_id)
- - update_quantity(product_id, new_quantity)
- - view_cart() → list all products with quantity and total price
- - get_total_bill() → calculate total bill of the cart
- - checkout() → Deduct stock from each product and empty the cart

4. Customer Class

Attributes:

- - customer_id, name, email, cart (ShoppingCart object)

Methods:

- - add_to_cart(product, quantity)
- - remove_from_cart(product_id)
- - checkout_cart()

Constraints

1. A customer cannot add more quantity than available stock.
2. Negative quantity is not allowed.
3. Cart should handle duplicate items gracefully (update instead of adding twice).
4. Once checkout is done, the cart should be cleared.

Examples:

1. Customer Alice browses the catalog.

- - Products available:
- - P101: Wireless Mouse (₹599, stock: 10)
- - P102: Keyboard (₹999, stock: 5)

2. Alice adds 2 Wireless Mice and 1 Keyboard to her cart.

3. Alice views cart →

Wireless Mouse x 2 = ₹1198

Keyboard x 1 = ₹999

Total Bill = ₹2197

4. Alice checks out.

- - Stock updated: Mouse → 8 left, Keyboard → 4 left.
- - Cart is now empty.

Implement this system using Python classes, objects and ensure that all constraints are respected.

DSP505: Programming Lab for Data Science and Artificial Intelligence

TPL616: Advanced Programming for DSAI

Lab-3: Practice Problems (Numpy)

Date: 03-Sept-2025

Instructions:

1. Try to complete lab problems during the lab hour and submit it through canvas. If you can't complete it within the lab time, you can submit it by the end of tomorrow.
2. Prepare all your solution files in a zip file and name it as <Name.zip> and submit on canvas.
3. You can use a jupyter notebook to solve the problems.

Problems Set-0:

Practice the codes given in the slides, execute them and make sure you get the correct output. No need to submit this.

Problems Set-1:

1. A list baseball has already been defined in the below, representing the height of some baseball players in centimetres. Prepare this as a numpy array.

   baseball= [180,215,210,210,188,176,209,200]

2. Create a numpy array from baseball list defined in Q#1. Name this new array np_height_in.
   a. Print np_height_in.
   b. Multiply np_height_in with 0.0254 to convert all height measurements from inches to metres. Store the new values in a new array, np_height_m.
   c. Print out np_height_m and check if the output makes sense.

3. A list weight_lb has already been defined in the Python script below, representing the weight of some baseball players in pounds

   Weight_lb=[172,190,165,175,166,159,182,168]

Create a numpy array from the weight_lb list with the correct units. Multiply by 0.453592 to go from pounds to kilograms. Store the resulting numpy array as np_weight_kg.

a. Use np_height_m (given in Q#2) and np_weight_kg to calculate the BMI of each player. Use the following equation:BMI=weight (kg) / height (m)^2
b. Save the resulting numpy array as bmi.
c. Print out bmi.

4. The code that calculates the BMI of all baseball players is already concluded in Q#3.Create a boolean numpy array: the element of the array should be True if the corresponding baseball player'sBMI is below 21. You can use the < operator for this. Name the array light.
    a. Print the array light.
    b. Print out a numpy array with the BMIs of all baseball players whose BMI is below 21.Use light inside square brackets to do a selection on the bmi array.e.g [light < 20]
5. In this question, baseball is a list of lists. The main list contains 4 elements. Each of these elements is a list containing the height and the weight of 4 baseball players, in this order. baseball is already given here for you

    ```
    baseball=[[180,78.4],[215,102.7],[210,98.5],[188,75.2]]
    ```

    a. Use np.array() to create a 2D numpy array from baseball. Name it np_baseball.
    b. Print out the type of np_baseball.
    c. Print out the shape attribute of np_baseball. Use np_baseball.shape
6. You've contacted FIFA for some data and they handed you three lists. The lists are the following:

    ```
    height = np.round(np.random.normal(1.75, 0.20, 1000), 2)
    weight = np.round(np.random.normal(60.32, 15, 1000), 2)
    age = np.round(np.random.normal(30, 10, 1000), 0)
    ```

    let’s stack it altogether

    ```
    np_players = np.column_stack((height, weight, age))
    ```

    Each element in the lists corresponds to a player. The first list, height, representing each player's height in metres. The second list, weight, represents each player's weight in kilograms. The third list, age,representing each player's age in years.

a. Create a numpy array np_height that is equal to the first column of np_players.
b. Print out the mean of np_height .
c. Print out the median of np_height

Problems Set-2:

Generate 2D NumPy random array representing sales data (units sold) of 6 products over 8 weeks. Compute the following.

a. Weekly Total Sales: Compute the total sales for each week (column-wise sum).

b. Best-Selling Product: Find the product (row index) that sold the most units in total.

c. Top 2 Weeks for Each Product: For each product (row), extract the indices of the two highest-selling weeks.

d. Products Above Average
   Compute the overall average sales (scalar). Return a Boolean mask of shape (6, 8) showing where sales exceed this average.

You can use the following numpy functions for solving.

sum, mean, argmax, argsort.

DSP505: Programming Lab for Data Science and Artificial Intelligence

TPL616: Advanced Programming for DSAI

Lab-4: Practice Problems (Pandas)

Date: 10-Sept-2025

Instructions:

1. Try to complete lab problems during the lab hour and submit it through canvas. If you can't complete it within the lab time, you can submit it by the end of tomorrow.
2. Prepare all your solution files in a zip file and name it as <Name.zip> and submit on canvas.
3. You can use a jupyter notebook to solve the problems.

Problem Set-0:

Practice the codes given in the slides, execute them and make sure you get the correct output. No need to submit this.

Problem Set-1 (Warm up exercises):

Download the Lab_sheet_Notebook.ipynb, corresponding dataset and solve the problems.

Problems Set-2:

Use the Titanic dataset to solve the following tasks:

Step 1: Load Data

- Load the `Titanic.csv` file into a pandas DataFrame.
- Display dataset info, shape, and first 5 rows.

Step 2: Handle Missing Values

- Fill missing **Age** values with the **median Age**.
- Drop the **Cabin** column (too many missing values).

Step 3: Detect Outliers

- Detect outliers in the `Fare` column using:
  1. **Z-score method** (|Z| > 3)
  2. **IQR method** (outside `[Q1 - 1.5×IQR, Q3 + 1.5×IQR]`)
- Create a new column `Outlier_Fare` = True/False.

Step 4: Use a Pipeline

- Create **functions**:
  - `handle_missing(df)` → handles missing values
  - `detect_outliers(df)` → detects outliers in Fare
  - `summarize(df)` → returns average Age, average Fare, and survival rate
- Apply them in a pipeline:

DSP505: Programming Lab for Data Science and Artificial Intelligence

TPL616: Advanced Programming for DSAI

Lab-5: Practice Problems (Matplotlib)

Date: 12-Sept-2025

Instructions:

1. Try to complete lab problems during the lab hour and submit it through canvas. If you can't complete it within the lab time, you can submit it by the end of tomorrow.
2. Prepare all your solution files in a zip file and name it as <Name.zip> and submit on canvas.
3. You can use a jupyter notebook to solve the problems.

Problem Set-0:

Practice the codes given in the slides, execute them and make sure you get the correct output. No need to submit this.

Problem Set-1:

Explore the different types of charts available in the matplotlib to visualize the health data as follows.

1. Load and Inspect Data
- Read health_data.csv using pandas.

2. Line Chart – Fitness Progress
- Plot monthly Steps and Calories Burned.
- Highlight months with the highest activity.

3. Bar Chart – Sleep Analysis
- Plot average Sleep Hours per Month using a bar chart.
- Add a horizontal line at 7 hours as a reference for healthy sleep.

4. Scatter Plot – Heart Rate vs Sleep
- Plot Avg_Heart_Rate against Sleep_Hours.
- Use color coding for Steps that month.

5. Pie Chart – Activity Contribution
- Compare total yearly Steps vs total Calories Burned (scaled).
- Show percentage contribution.

6. Subplots (2x2 Layout)
- Combine line, bar, scatter, and pie charts in one figure.
- Add the main title: 'Health & Fitness Analysis (2024)'.

7. Save Report
- Export the figure as health_report.png.

Problem Set-2:

Load iris dataset as follows.

```
import seaborn as sns

iris_df = sns.load_dataset('iris')

iris_df.head()
```

Perform the following tasks:

1. Plot the relationship between PetalLengthCm and PetalWidthCm using a 2D scatter plot.
2. Now, plot a 3D scatter plot by adding SepalLengthCm as the third dimension.

Which plot (2D or 3D) is more effective in helping you distinguish?

DSP505: Programming Lab for Data Science and Artificial Intelligence

TPL616: Advanced Programming for DSAI

Lab-6: Practice Problems (Seaborn)

Date: 16-Sept-2025

Instructions:

1. Try to complete lab problems during the lab hour and submit it through canvas. If you can’t complete it within the lab time, you can submit it by the end of tomorrow.
2. Prepare all your solution files in a zip file and name it as <Name.zip> and submit on canvas.
3. You can use a jupyter notebook to solve the problems.

Problem Set-0:

Practice the codes given in the slides, execute them and make sure you get the correct output. No need to submit this.

Problem Set-1:

Download the heart disease dataset from the Canvas and perform the following tasks. The dataset contains the following columns.

Dataset
- age: Patient’s age
- sex: Gender (1 = male, 0 = female)
- cp: Chest pain type (categorical: 0–3)
- trestbps: Resting blood pressure
- chol: Serum cholesterol (mg/dl)
- thalach: Maximum heart rate achieved
- target: Heart disease presence (1 = yes, 0 = no)

Task 1: Pairplot
- Create a pairplot for the columns ['age', 'chol', 'thalach', 'target'].
- Use hue='target' to differentiate between patients with and without heart disease.
- Write a short explanation of any visible separation or clustering.

Task 2: Boxplot
- Draw a boxplot of trestbps (resting blood pressure) grouped by target.
- Add another boxplot for chol grouped by sex.
- Interpret whether there are differences in distributions.

Task 3: Violin Plot
- Plot a violin plot of thalach (max heart rate) split by target.
- Add another violin plot for age split by sex.
- Compare the spread and medians across groups.

Task 4: Heatmap
- Compute the correlation matrix for all numerical features.
- Plot a heatmap with annotations.
- Identify which variables are strongly correlated with the target.

Task 5: Jointplot
- Create a jointplot of age vs thalach with hue='target'.
- Try both kind='scatter' and kind='kde'.
- Explain whether younger or older patients tend to achieve higher max heart rates.

Task 6: Barplot
- Create a barplot of cp (chest pain type) vs average thalach, with hue='target'.
- Create another barplot for sex vs average chol, with hue='target'.
- Discuss differences across groups.